LINKING TWO BUILDING PERFORMANCE SIMULATION TOOLS TO A PRODUCT MODEL TESTBED

Godfried Augenbroe, Pauline Wilschut and Wouter Rombouts
Department of Civil Engineering
Delft University of Technology
P.O. Box 5048, Delft, The Netherlands

ABSTRACT

The development of an interface, which links two building simulation tools to a test version of a product model is discussed.
The two simulation tools in hand can be regarded as representative members from the broad spectrum of building performance evaluation (BPE) tools:

- BFEP: a component-based program for the simulation of the temperature behaviour of buildings
- SIBE: a program for calculating the solar irradiation in the built environment.

After dealing with the concept of a Product Model as the the complete and consistent representation of the building, the functionality of ProMod, which was developed by TNO-Bouw in the Netherlands as a testbed for product modelling concepts, is briefly reviewed.
Starting from the analyses of the data requirements of the two BPE tools (in our case resulting in IDEF1X data models), we show how both data models can be mapped to one single integrated data model, and demonstrate how this data model is implemented in ProMod.
The actual interface (programmed in C) retrieves all the building data from ProMod and prompts the user for additional application-specific data. An intermediate file is produced with the complete input required by each BPE-tool.
With a view towards future full blown data integration we discuss the demands on future developments.

1. INTRODUCTION

One of the main themes of present building simulation research is the integrated use of available tools in a building design context.
One usually distinguishes different increasing levels of operational integration, e.g.

- Some links between tools are established, i.e. output from one tool is easily (without human intervention) transferred as input to another tool.
- All tools share a common description of the "real world object", they all deal with. Based on this description they can exchange all relevant information about the object, be it in some specified neutral format or through common access to a database that holds the description of the actual object. We will call this type of integration "data-integration".
- The data-integrated tools are harnassed in a software-architecture, thus adding increased functionality (e.g. for design purposes) to them. We call this added functionality process-integration.

We will focus on the second level of integration and present how two BPE (Building Performance Evaluation) tools have been linked to a common description in the form of a common data model of a building.
Future integrated systems will be based on one comprehensive object description, accomodating all information needs from many different parties that take part in the different stages of a building project. Such a complete and consistent set of data about the building is called Product Model; it must contain data about all aspects (costs, strength, color, etc.) and functions, be able to represent the shape and structure (topology and geometry) and be able to record the different stages that the object goes through (from conception to demolition, if you like).

In order for such a product model to support open exchange of data between "non-integrated" partners in a building project, a common description of a building will sooner or later have to be standardized.
The ISO-STEP-standard is a first step towards that goal (ISO 1989).
In order to narrow down the scope of our discussion, let us consider the following global tasks in the development of a product model:

task 1 define an overall framework for the description of a product (Danner 1990).

task 2 integrate different views and representations of the product as required by the different actors that want to exchange their particular "views". Define mappings which relate the common description entities to these local view entities.

task 3 each actor formalises his view on the product in a suitable formalism and submits it to task 2.

Task 1 is essentially a top-down effort, performed by specialists, whereas Task 3 is a bottom-up effort, to be performed by the different actors in the building process. Task 2 is the task where all efforts come together and the common description will gradually take shape.
We will concentrate on task 3 showing how data models are developed for two particular BPE-tools.
In no way will we attempt a general discussion of the development of future building type product models. Rather we will demonstrate a road which must be explored by BPE-specialists in order to make their tools available to the task 2 efforts. The key issues of developing building data models are explored to the extent how they bear down on the data modelling of individual tools.
The views expressed are to a great extent forstered by the work done at the TNO-Building and Construction Research Institute. The actual interface was developed using the TNO-ProMod Product Model tested (Kuiper et al. 1987).
It must be noted that TNO has recently begun to develop a more flexible successor to ProMod. Although ProMod lacks the features of a production tool and does not support all desired modelling concepts it has proven to be an excellent exploratory tool for our purposes.
We will first show the data models that were developed for two particular BPE-tools.
We will then go on to demonstrate how the two data models can be integrated, and discuss this exercise in a general framework.
We will then briefly discuss the actual implementation of an interface between two BPE-tools and ProMod.
In the following sections we will also touch upon crucial data modelling aspects from the BPE point of view, as they determine to a great extent the success with which BPE-tools can be interfaced to future product models.

2. TWO PARTICULAR BPE-TOOLS

2.1. SIBE

The first BPE-tool considered, is SIBE (Solar Irradiation in the Built Environment) which calculates solar irradiation and daylighting data on arbitrary planes in arbitrarily shaped building configurations (v.d. Voorden 1985; v.d. Voorden 1988). SIBE consists of several modules, each of which is dedicated to a particular "solar" aspect. We will only deal with the SIBE1-module, which addresses direct solar irradiation. The following list of functions specifies capabilities and provides an informal survey of the data that SIBE1 handles:

- the environment (i.e. all solar obstructions) is specified in a hierarchical way, i.e.:
  - . the environment consists of a group of buildings.
  - . each group consists of a number of buildings.
  - . a building can have building parts.
  - . a building part consists of a floorplan-perimeter and a height attribute.
    Note that all faces of the building are thus implicitly defined.
  - . each perimeter is identified by a number of "perimeter points".
  - . additional faces can be specified; they can be subdivided in "related" (room envelope walls) and "non-related" (e.g. overhang) faces.
  - . perimeter points are identified through their coordinates in either local or global coordinate systems. Local coordinate systems can be specified on seperate levels of the hierarchy (not shown in the diagram).
- window-openings are identified by a number of perimeter points.
- a "plane of interest" can be positioned arbitrarily; SIBE1 calculates irradiation data for a number of grid-points on the specified face-side of the plane. The outer perimeter of the plane of interest is specified through a number of perimeter points and grid-size parameters. Examples of what purposes the plane of interest can serve in typical SIBE1 calculations:
  - . a children's playground (to investigate direct sunshine hours)
  - . a window, or an internal wall in a room, with a window (energy/comfort analysis)
  - . a solar roof collector (solar energy analysis)

If the plane of interest is located in a room the envelope of the room as well as additional obstructions inside it must be specified as "related" additional faces, in which window openings can be positioned.

Notice that, at least semantically, the difference between "related" and "non-related" faces is purely a matter of where the plane of interest is located, i.e. what room one is dealing with. This is to be regarded as a typical local application-flavor, because the calculation procedures inside SIBE1 make efficient use of this distinction. As we will discuss later, it turns out that these local flavors are hard to support in a

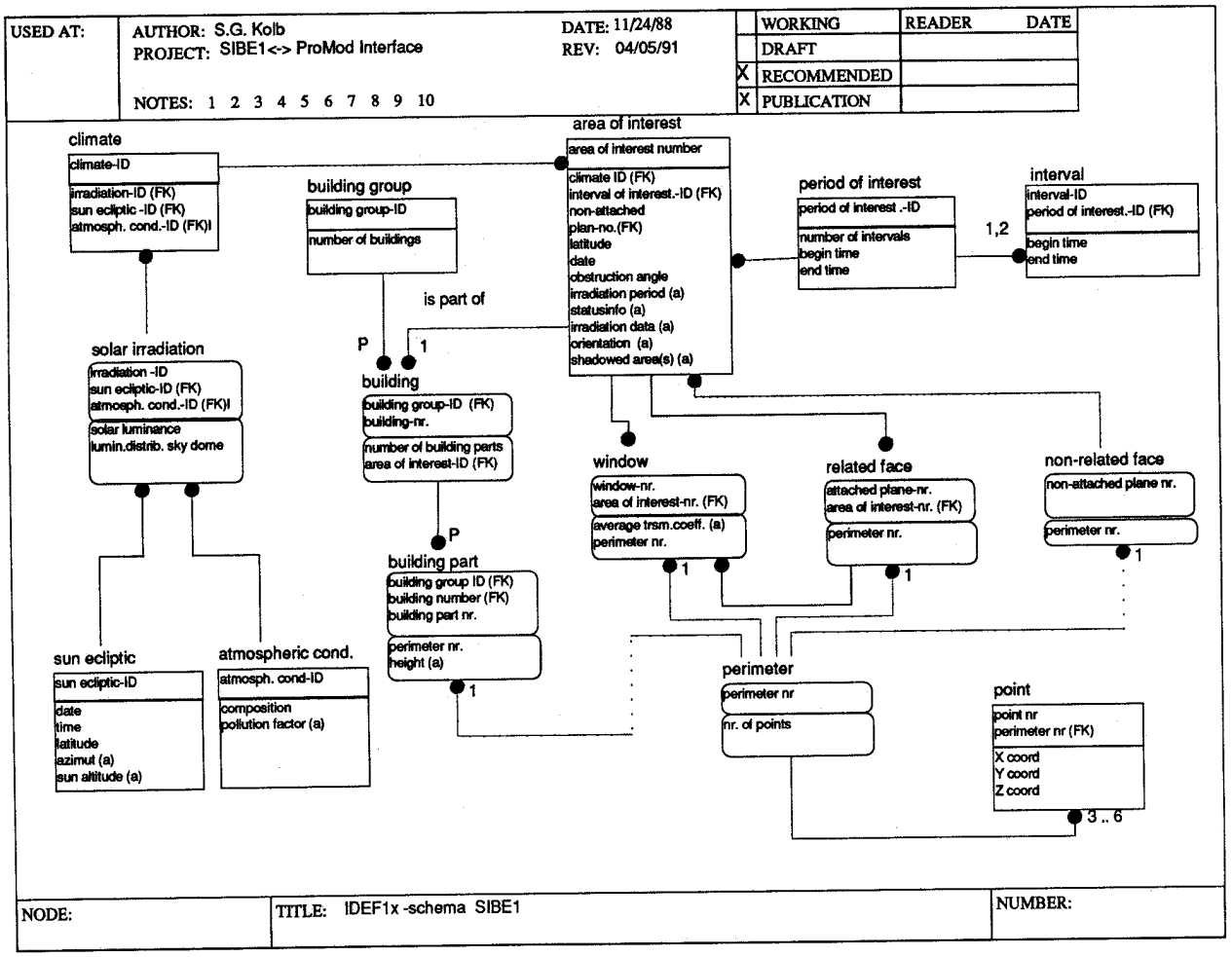

Fig. 1. IDEF1X-diagram of external data of SIBE1 (Output Data not shown).

general product description. How this problem is dealt with (or rather worked around) is explained later on.
- Other global data are concerned with data, location on earth, dayly intervals of interest.
- In order to translate direct sun exposure intervals to incident radiation, relevant climate data, i.e. atmospheric conditions must be specified as well.
- Output data concerns:
  . direct exposure intervals: for each grid point, dayly intervals are stated.
  If the plane of interest is inside a building and there are multiple windows, exposure intervals attributed to a particular window are shown.
  . design information: for each building an affect-status number is produced, showing whether or not the building affects the exposure of the plane of interest.
  . incident energy flux: using the solar intensity attributes, an integrated value (in time and space, i.e. summed over the grid points) is presented per time step, due to direct sun.

Some additional remarks about SIBE1:
- The description above is confined to the so-called external view of the BPE-tool, i.e. dealing only with input and output data.
- Internally SIBE1 uses a quite unique approach, i.e. not based on traditional ray-tracing techiques, but employing powerful and efficient algorithms based on analytical solution techniques for determining the "intersection points" of the sun ecliptic and any obstruction.
- Although the external view normally regards the algorithmic approach and its related entities as being private to the BPE-tool (an actual case of data hiding), more often than not the external view contains the "finger prints" of the application algorithms. In fact this "bias" of the external view poses a number of difficult problems for the integration of these external views in a generic building data model (section 1, task 2). The case of "related/non-related" faces is a perfect example of this.

In figure 1, the external view of SIBE1, i.e. data model of the external SIBE1-data is shown. The data model is represented in the IDEF1X-modelling formalism (Gale 1986).

The diagram of figure 1 is the starting point for our integration, or rather interface-exercise which we will discuss in section 4.

2.2. BFEP

BFEP is a toolbox for temperature/energy/comfort calculations in buildings (Augenbroe 1986).
It's backbone is a set of finite element-based space discretization functions and a set of numerical time integrators.
BFEP-use falls in three categories:
1. On the lowest level a library of elements and a set of standard routines for performing standard finite element technique operations is supplied. A specialist user can incorporate these functions in his own software.
2. Non-specialist users will rely on a set of pre-defined typical building components and operations (wall, room, collector, ventilation, ...). The user can compose his system by choosing appropriate parametrized components and specify their interfaces (actually also dealt with as components by BFEP). In fact, the assembly hierarchy is transparent to the user, as components consist of assembled elements and components can be assembled to form macro components and so on. Operations on the components (in BFEP-terminology the process-oriented behaviour) have to be specified still in user-supplied software.
3. For most convenient use a software program (a so-called BFEP-driver) is configured based on the basic toolbox-functions, but with hard-wired behaviour, offering only a limited set of options. A BFEP-driver releaves the burden of writing additional software, but is obviously limited to the application on a set of pre-defined problems.

We will now describe Driver-1, which is a particular example of existing BFEP-drivers; it has severe limitations in its applicability but it is simple to use and serves our interfacing purposes well. Moreover, for the sake of brevity we will only deal with one option of Driver-1: cyclic behaviour of the room temperatures for a user-specified "average day".
A brief review of the capabilities of Driver-1 and its external data is given below:
- BFEP uses a particular aggregation hierarchy, solely based on unique node-numbers (finite elements nodes); which in effect could be regarded as unique labels of external state variables of a component. Components are implicity connected by sharing the same external state variable (components have also internal state, which is private and can never be shared). Remember that interfaces between components are regarded as components themselves.
  Components have parametrized local geometry, which is used to define its shape, in one, two or three dimensions. Standard routines discretize the shape in finite elements.
  The BFEP assembly-structure, i.e. the topology of the building is thus of an abstract nature, based on unique nodes to which components are attached. This is another example of how an application view dominates the way in which the system is defined. In this case physical relations (private to component behaviour) and exchange between components have driven the choice of a topological structure.
  As it turns out, the mapping of an actual building by a human modeller to fit a certain model abstraction is aided to a large extent by this abstraction mechanism!
  The challenge is then to formalize and automate this mapping from a general topology/geometry description like RM-rep, as explained in section 3.
- Basically then, the static part of the Driver-1 input (i.e. object description) consists of a list of components, implicitly referencing each other by having common external states.
  External states are identified by the user through unique node numbers. Examples of external states are: room air temperature, surface temperature, outside air temperature, thermostat set points, etc.
  Each of these is assigned a unique node number by the user in the physical modelling phase.
- Driver-1 puts severe restrictions on type and number of components that may be used: only one room, two windows, any number of walls and ventilation are the basic components allowed.
- The other process-oriented (i.e. dynamical) part of the input consists roughly of:
  . climate conditions of a design day: place one earth, date, hourly values of weather conditions
  . internal load conditions
  . cooling/heating set points (specified for user defined time intervals of the average day)
  . change conditions of system parameters (an input value for a component), specified for user-defined time intervals of the average day.
- Driver-1 calculates all hourly temperatures in all nodes, additionally hourly heating and cooling loads for the room as well as the mean radiant temperature inside the room are calculated.

Figure 2 presents the IDEF1X diagram of the external view of Driver-1.

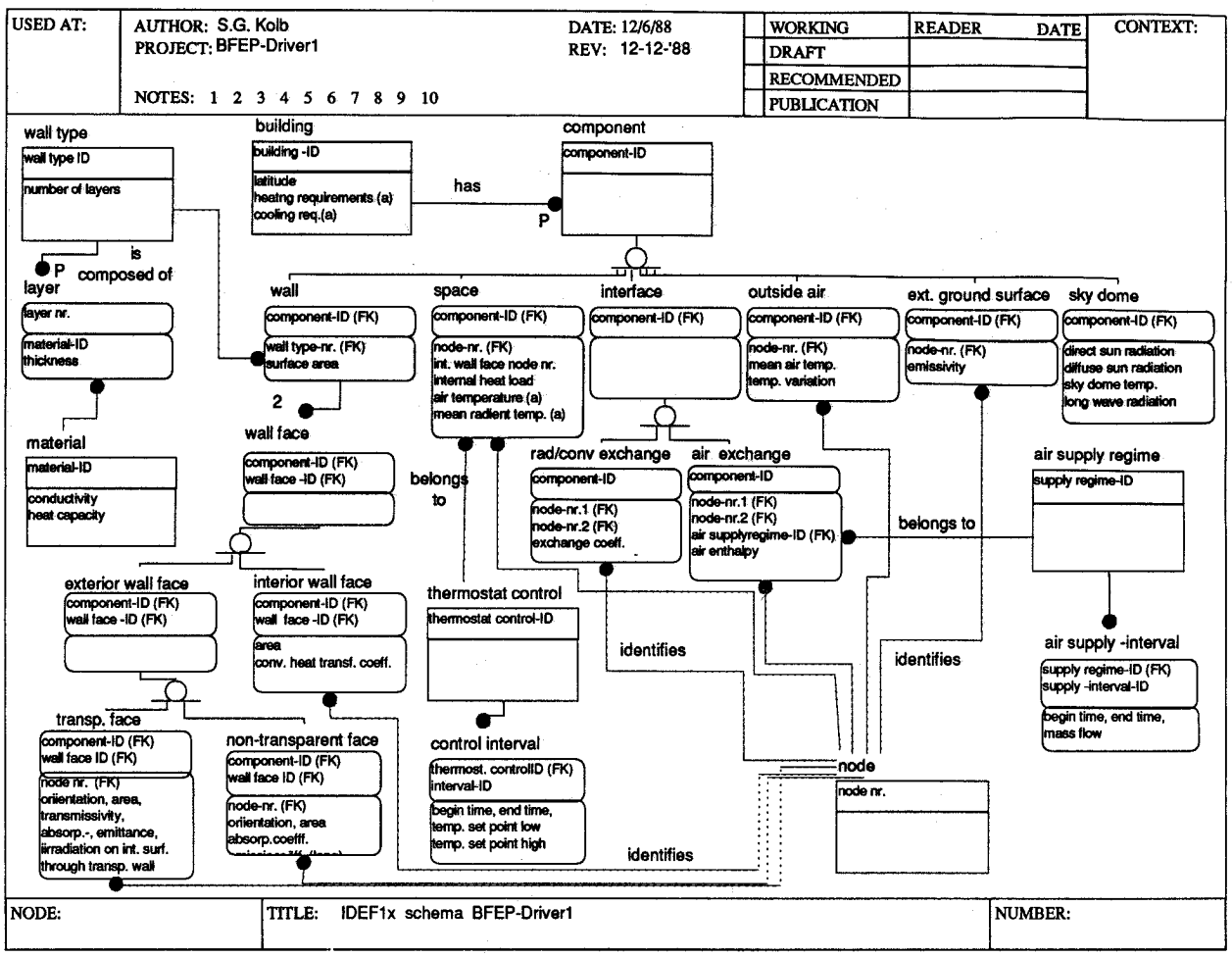

Figure 2. IDEF1X diagram of external data of BFEP-Driver-1

3. KEY-ISSUES IN BUILDING DATA MODELLING

As explained above, the scope of the data modelling effort is determined to a large extent by the views we want to integrate. If we constrain the effort to the two particular BPE-tools of the previous section we are confronted with a fairly simple exercise. But realizing that the views of the two BPE's differ considerably, even such a small exercise turns out to be rather non-trivial. In order to accomodate the two application-biased views in hand leaves us with the task of finding the right generic entities from which both views can be generated or derived.

Application-specific interfaces must then be developed to take care of the actual mapping between the resulting integrated data model entities and application entities.

Other functional requirements would further guide implementation choices, such as whether to exchange through a neutral file (static interface) or through a shared database access (dynamic interface). Some of the topics that guide the development will be discussed below.

3.1. Framework and modelling paradigms

In general, any data model development effort will be guided by the following considerations (Tolman 1989):

- the intended scope of the central data model; i.e. what data is exchanged through the central model, and what data is considered to be application-specific and thus "private" to a BPE.
- what top-down concepts and abstraction mechanisms are supported by the data model, e.g.
  - . generalization-specialization: in order to reduce data redundancy we need a mechanism to "describe" entities on different levels of specialization, e.g. employing a generic ("an air duct"), specific ("a PVC-airduct with diameter 0,15 m"), occurrence ("the PVC-airduct, diameter 0,15, length 6.00 m between joint A and joint B") hierarchy.
  - . aggregation-characterization: entity-description should support "aspect-of" relationships, thus enabling an aspect-oriented (e.g. color, strength, cost) view of the product.
  - . decomposition-composition: the data model should support "is part of" relations, thus enabling views on different levels of aggregation.
  - . life-cycle stages: a building description goes through life cycle stages, that could be classified according to some process-oriented categorization of specific decision-points in time.

    As integration across life cycle stages is one of the prime targets of integration, the data model must support life cycle views and their relations.
- the availability of a general framework to harmonize ongoing data modelling efforts. In fact the ISO-STEP effort performs a major role in this respect; a recent STEP-document defines the following layers (Danner and Yang 1991).

. definition: all data-aspects of a product, other than its shape-representation.
This layer in fact contains three sub-layers, dealing with context-definition, product-definition and property definition.
Example: for a duct system component definitions and their relations would be on this level.
. shape representation: all data-aspects of the shape of the product.
Example: representation of the shape of a duct, e.g. entity "cross-section" with attributes "cilindrical" and radius-value.
. shape presentation: all data aspects of how the shape is presented.
Example: the duct-cross section could be presented by either center point and radius or through 3 points on the cilinder-boundary.
It should be noted that the presentation layer is strictly speaking not considered part of the general framework of STEP.

A few observations are in order to highlight some of the challenges that one faces in the development of complete product models:

- no complete building data model will become available in any foreseeable future.
It is therefore important to build in guarantees for future extensions in ongoing limited developments. Support of above mentioned concepts is one of them.
- as different application views imply different levels of abstraction, it is of the utmost importance to support useful abstraction mechanisms. It is debatable whether characterization and decomposition supply enough richness in this respect. Often a particular abstraction is guided by an application-specific schematization and (physical) modelling approach.
Further work in the area of modelling methodology based on a categorization of physical agents and a taxonomy of building behaviour should blend in with ongoing data modelling efforts. Among others, valuable input is expected from the EKS project (Clarke 1988).
- one must realize that important operational and user issues of the (implemented) central data model are more or less blocked out from present day attention, e.g. ownership, versioning, authorization etc. In fact, especially the general support along the life cycle stages-axis will entail a great deal of implied procedural knowledge dealing with these issues. Process knowledge captured in process and decision flow diagrams will thus have to supply essential support to the use of the building data. Although we see rapid progress in CIM projects, mostly dealing with fairly simple part production processes, application to the building industry seems to be some decades away.
- another important issue is the type of product modelling power one is able to supply to different types of users, i.e. "Who can define what?" Taking an existing first shot at a building product model supplied for instance by an ongoing data modelling exercises, it does not take too much imagination to come up with an existing building that one would not be able to model according to this data model structure.
This fact would make it quite unacceptable to the average designer, one suspects.
So, why not take a broader view and only define the way we describe the product and not the product itself. This is exactly what an important result from the STEP-community (Gielingh 1989) is about. This would imply that a major part the real modelling power, i.e. what the product IS and not just filling in a "hard-wired" description format, is delivered to the designer.
Obviously, for any meaningfull exchange of data to occur, one would have to adhere to certain conventions known to both transmitting and receiving ends, which however would not have to be hardwired if one can find a way to exchange these conventions on the fly.

3.2. Topology-geometry issues

Since the earliest introduction of CAD systems, design has been dealt with as a visually driven exercise. In fact the emphasis on drawing capabilities of present CAD systems is still pre-eminent. Recent years have witnessed a quite natural evolution from presentation-based "drawing" systems to more design-oriented systems enabling the addition of design semantics to a shape representation, the latter still being regarded as the backbone of the design process. Interestingly enough, growing requirements on capturing the semantics of a product are pushing the shape aspect even more to the background, as is reflected by the STEP-framework mentioned in the previous section.
To elucidate this, let us reflect on the different roles that the shape-aspect can play:
- in present day CAD-systems: a shape-driven design approach; topology and geometry are used as the "glue" by which entities are connected. A closer look reveals that this approach can be very restrictive and is in fact unable to support a top-down design process. Such a process requires that many relations (some of which topological by nature) can be applied on different refinement levels, even before some shape information is available.
- in future CAD-systems: relations are explicitated on different levels of refinement, topological relations (e.g. connectivity relations) are refined on lower levels adding geometry where appropriate. This implies that shape is (partly) evaluated from semantical definition of the product. In other words, the product glue is in the product description itself, not in its shape.

As far as classical data models for shape representation are concerned we conclude that they do not as such supply us with enough power to support the second approach.
Looking more closely at these data models (fig. 3) we can make a few general observations (Tolman 1989):
- Wire frame representation (WF Rep): the simplest representation, dealing only with topological Vertex and Edge entities. Geometry is nicely seperated through adding Shape Type to edges and Coordinate Type to vertices.
It needs little clarification that this model is inadequate for a general building shape representation as no knowledge about spaces (voids) and solid material can be stored.
- Boundary representation (B-rep): a popular solid model representation dealing with topological Solid-Face-Edge-Vertex entities, whereas geometry is added similarly to the WF-rep.
Although much richer that WF-rep, B-rep is still too restricted for building modelling as it is unable to model internal spaces, which is not surprising in view of its solid model origin.
- Reference Model representation (RM-rep): based on B-rep with added Cell and Loop entities in order to adequately model spaces and holes (voids) in faces. Face Side is introduced to unambiguously determine the relation between two adjacent Cells.
Again geometry is not shown; all depicted entities are topological, which thus act as the reference for the actual shape represented in geometry entities (this aspect explains its name Reference Model, i.e. a topological reference structure for the explicit geometrical shape). It must be noted that for building shape representations on a non-detailed level (e.g. regular spaces, flat-walls) no extra geometry-entities are needed.

Although the RM-rep is quite adequate to model the shape of buildings it cannot be used as the spinal cord of the building product model.
The main reasons for this will be summarized below:
- in the early design stages, one must be able to store information although there is as yet no explicit shape information.
- different abstraction levels should be supported by different shape representations.
As we have seen in section 3.1, the support of explicit

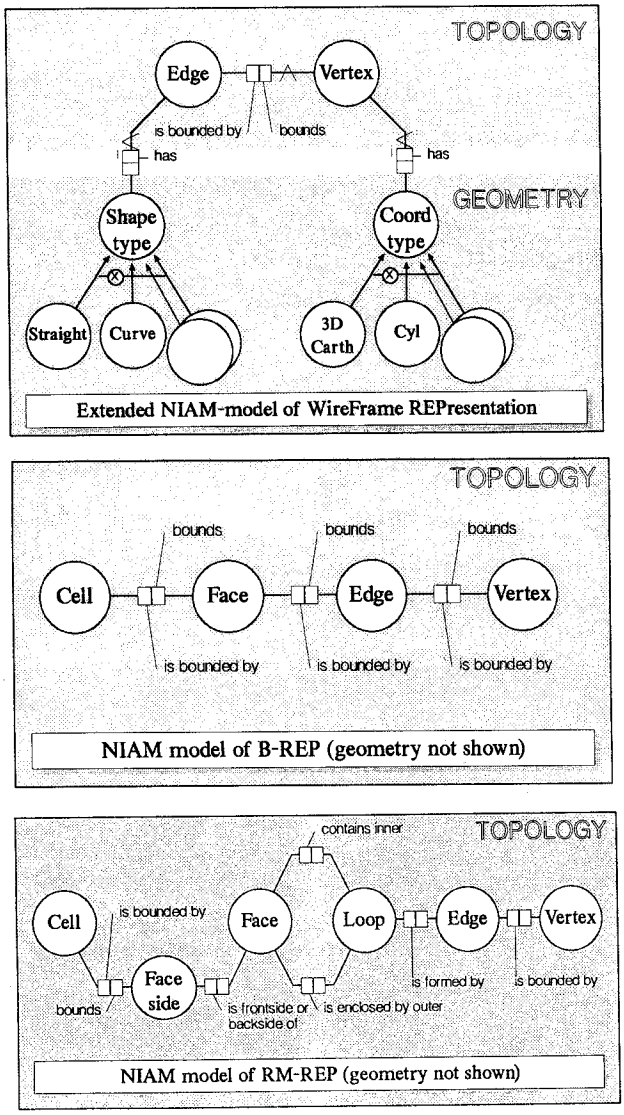

Figure 3. Classical shape representation models in NIAM format

abstractions is of key-importance to support different views of a building; it is quite obvious that these abstractions require different levels of shape representation detail, in order to support application models of a building.
If this requirement is not fulfilled, the totally "flat" shape description spans the range from global objects (room, wall) to the finest details (door knob) on one and the same level.

3.3. Availability of solutions

All requirements of the previous sections could not and obviously need not be fulfilled in the very limited exercise in hand, dealing with just two BPE-tools, resulting in a minimal set of aspects and moreover posing no demands on life-cycle support.
So, the solution for our integration exercise based on the product model testbed ProMod explores the limited use of a limited set of abstraction mechanism.
Also the representation solution that we used, is based on some compromises with respect to the ideal case, described in the previous section.

Acknowledging that ProMod is a research rather than a production tool, we argue that the market does not provide us as yet with tools that would support our demands. This underlines the fact that solutions as yet exist only in the research laboratories, on paper or at best in prototypes.
We agree that there is still quite a long way to go before these type of product modellers will have established a place on the market for themselves.

4. INTERFACE DEVELOPMENT

The actual interface will now briefly be described; full details can be found in (Wilschut 1990).
First the main characteristics of the product model testbed ProMod are highlighted, followed by a discussion of the combined data model for SIBE and BFEP as it is implemented in ProMod. Due to space limitations, the actual interface program is touched upon only briefly.

4.1. ProMod: a product model testbed

ProMod is a development by TNO-Bouw in The Netherlands (Kuiper et al 1987, Krom 1989). It was developed to test product modelling concepts, as they are being developed by the same research group.
Most notably, the modelling paradigm in the general AEC Reference Model (GARM) has found a good testing ground in ProMod.
It must be noted here that the ProMod version we used for the interface development will soon be replaced by a new one. This new version will provide broader support for all GARM mechanisms, as well as support for several new modelling mechanisms, especially in the area of support along the composition-decomposition axes with respect to matching shape representation. Important theoretical work in this respect, e.g. generalized topology and meta-topology can be found in (Tolman 1989).
ProMod, as we used it, runs in a UNIX and C environment and uses GKS for graphical presentation. A customized database SDS, which holds the actual data, can be accessed through a number of specific tools to interactively present the shape of the model or query data from the database. Dedicated interfaces can be built by using data access-functions, callable from a C-program.
ProMod provides full support for the RM-rep shape representation, whereas solid models and 2-D drawing presentations can be generated from the RM-rep representation.

User-interface: the user interface in the employd version of ProMod deserves some elaboration, it works through a Lisp-style interface tool, called XPCL (XLisp based Product Model Construction Language, an extension of XLisp) (Krom 1989).
Exploiting the object oriented mechanisms offered by XLisp, the user can define his model by instantiating objects, naming them, giving values to attributes, and referencing objects to other objects, all of which is done by sending appropriate messages to objects. XPCL provides a predefined set of object classes with a specified hierarchy, along which the inheritance mechanism is supported. The complete set of XPCL-source must be submitted in a file which is then interpreted resulting in ProMod routine calls to instantiate the product model in the database. Although suitable in a research environment, this way of working brings several serious limitations to mind for practical use. The following versions will presumably address this problem satisfactorily.

Abstraction mechanisms. The two abstraction mechanisms that are the core of the GARM and are to a considerable degree supported in the present version of ProMod are staged below:

1. FU-TS decompostion. Starting from the following notions of the generic entities: Functional Unit (FU) and Technical Solution (TS), a decomposition paradigm based on a decomposition in FU's matched with TS's is adhered to (Gielingh 1989).
   A complex (topdown) design problem is supposed to be decomposable into smaller ones by a decomposition where each FU (stating requirements) is matched by a TS (having characteristics). Each TS can be decomposed in a set of requirements (FU) on a lower decomposition level. An example of a decomposition will be presented in the next section.

2. Specialization-Generalization:
   For both FU and TS, ProMod supports three levels of specialisation. For FU's this is implemented in the following three classes:
   - GenFu: Generic Functional Unit
     Description on a general level
   - SpeFu Specific Functional Unit
     (Parametrized) description on a more specific level
   - FUOcc Functional Unit Occurrence.
     Completely determined instance of SpeFU, i.e. parameter values are set and position and orientation in space are fixed.

   The same specialisation is also used for TS (i.e. GenTS, SpeTS, TSOcc). References from FU to TS can take place on several levels, depending on what is most appropriate in given circumstances. For instance: referencing a TS to a SpeFU would result in the same technical solution for all FUOcc of that SpeFU, whereas referencing a TS to a FUOcc, would relate the technical solution to just that one occurrence.

Having established the way we can exploit the two main abstraction mechanisms for our purposes, we are now confronted with finding the best way to adapt them to our data modelling requirement, i.e. integrating the SIBE and BFEP data models.
This will entail managing the decomposition and specialization mechanisms in the best way to arrive at a flexible and easily accessable definition of the "product", which in the limited scope of our project is merely the sum of two application views.

A few additional comments on the use of ProMod are in order here to finish up this short account:
- Although ProMod + XLisp offer extensive capabilities for shape representation, it should be emphasized that decoupling representation from decomposition (an ideal requirement, stated in the previous chapter) is not fully supported in the present version. In that sense the topological "glue" is still supplied by linking the topological representation of smaller entities, i.e. a new TS is made through performing a join-operation on the representations of its decomposed FU's. This has the effect that, as FU's are grouped together in a TS which is one level up, the connection of FU's is actually through the linkage of their representations.
- ProMod is equipped with a label-reference construct which allows labelling technical data to a TS (e.g. physical data of a wall-layer) and labelling functional data to a FU (e.g. functional requirements of a room).

4.2. Data model implementation

The description below is intended to give a general idea of the data model as it is implemented in ProMod. Many details will be skipped in view of space-limitations.
The diagrams will present the general structure of the model in terms of object classes and how they reference each other, thus staying closely to the XPCL structure through which the user implements the data model.

Only the end result of the integration exercise, i.e. the data model that combines both SIBE and BFEP views, is presented. We argue that there is no clear methodology which leads to an integrated data model and it is this aspect that requires thorough inspection before embarking on a a large integration effort.
But, in the case presented here, the general structure of the model was more or less self obvious from the start, requiring merely some refinements along the way.

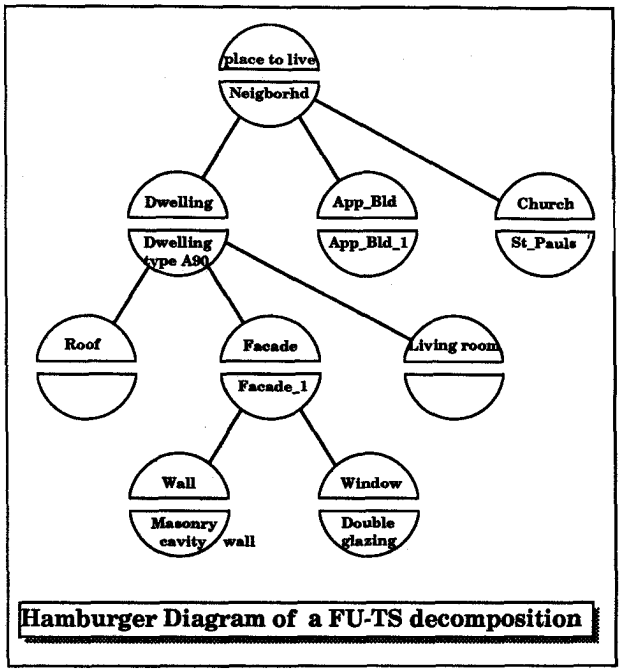

Fig. 4. FU-TS decomposition of implemented data model.

Fig. 4. presents the general FU-TS decomposition which has proved to be adequate for our purposes. The choice of decomposition levels has been driven by the two BPE's considered.
As such, the decomposition will be incomplete and accidental in some respects, thus showing the effects of the limited bottom-up approach. The decomposition in fig. 4 is pursued only in part in order to demonstrate the essential levels. It must be emphasized that use is made of implicit functional descriptions, i.e. "facade" thus signifies "function of any facade" and "window" likewise stands for "function of any window".
It must also be noted that the decomposition is very biased by the two BPE-views; as a decomposition of a window in a window frame and glazed parts is apparently not required by any of the two BPE's, this decomposition is completely absent and the shortcut to the type of glazing is the only one present. Frame-width is obviously neglected or taken to be a default percentage of total window area by the two BPE's.
Adhering to this specific decomposition, we are now able to instantiate a particular neighborhood, consisting of one dwelling, one appartment building and one church in a product model, which is named: "Neighborhood".
This is demonstrated in fig. 5 where again only the dwelling is (partly) shown.

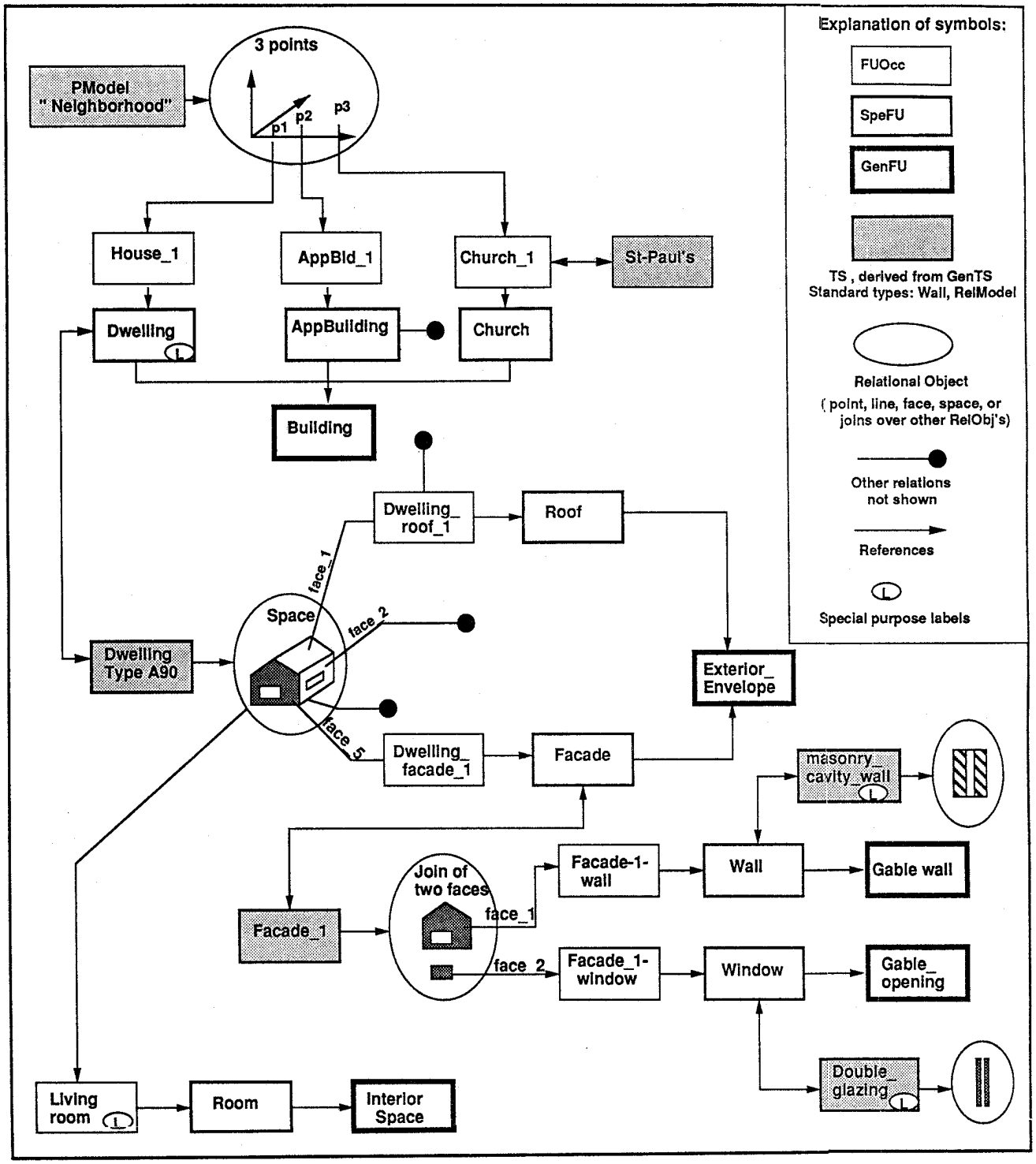

Fig. 5. Implemented data model (XPCL-structure).

Fig. 5 follows the structure of fig. 4, whereas relational objects to represent shape are added along the way. Also generalization is pursued on different levels, by adding SpeFu and GenFU objects.
Full exploitation of this mechanism is not really at stake due to the limited scope of the integration effort.
Some observations must be made:

- as explained earlier, the FU-TS reference can take place either on FUOcc-level or on SpeFU-level. Because of the low demands on generalization in the case in hand, choices in this repect are mader more of less arbitrarily. The TS for "Dwelling Type A90" for instance is related to the SpeFU "Dwelling", supposing that the neighborhood consist of many dwellings of the same type, thus differing only in

place and orientation. Each occurrence (e.g. House-1, referring to a place/position P1) would reference this particular dwelling type through its SpeFU "Dwelling".

- Fig. 5 assumes that a dwelling consists of only one space ("Living Room"), which is in accordance with fig. 4, where only one function of a dwelling is distinguished.
  Introducing more functions, and thus more spaces, an additional decomposition level would have to be introduced. As such this is not difficult to realize, but due to specific BFEP-requirements the interface for this case has not been realised as yet.
- The Relational Model for the dwelling representation consists of a Space object. It should be noted that the FUOcc "Living Room" references the Space as such, whereas FUOcc "dwelling-roof-1", "dwelling-facade-1" reference individual Faces of Space. This exemplifies the earlier stated fact that decomposition is enforced through the topological structure of the shape-representation, which, in the ideal case, we would rather not have.

4.3. Interface implementation

Having instantiated a particular neighborhood in ProMod according to the structure of figures 4 and 5, we need to interface both applications to ProMod.
This is accomplished through a static interface, written in C, which queries the ProMod database and prompts the user for additional data input.
The interface program produces complete input files for SIBE1 and BFEP. At the present stage of development, output data is not fed back to ProMod, which of course would be required for full integration of the two applications. In that case irradiation data produced by SIBE1 would be available as input to BFEP for temperature calculations. At this stage BFEP uses a reduced set of SIBE-modules to do its own irradiation calculations.
The present version of the interface program behaves according to following scheme:

- speficy the name of the ProMod model
- specify whether input for SIBE1, BFEP or both is to be generated.

For SIBE1 the user is prompted to:

- choose the buildings that enter the calculation
- position the plane of interest and supply grid size parameters.
  . if the plane of interest is inside a building, this building must be specified; the interface program will collect all faces of the building as "related faces" and collect windows that can affect the plane of interest.
  . if the plane of interest is exterior, all faces of all selected buildings are collected as non-related faces; windows are disregarded.
- supply application specific data, such as date, dayly-intervals, latitude, climate conditions. The latter refer to data private to SIBE1.

For BFEP the user is prompted to:

- select the room for the temperature calculation; this is identical to selecting a building, because at the present moment no decomposition of a building in seperate rooms is supported. Through the added functional data in ProMod, referenced by the FUOcc of the room, the interface selects appropriate data for thermostat set points, convective and radiative heat exchange coefficients, room capacity, etc. The latter data is private to BFEP.
  The interface collects all walls and windows which constitute the envelope of the wall.
  Physical data of wall layers and glazing are accessed through the labels supplied in ProMod to Technical Solutions.
- Application-specific data such as integration time step, climate conditions for the average day (through choice of set of pre-defined options), air-supply regime and control intervals.

We conclude this short account of the interface by making the following observations:

- the description of a neighborhood must follow the strict decomposition levels of fig. 7, as the interface expects to encounter walls and windows on the pre-defined decomposition level. Moreover, in order to detect windows, strict naming conventions must be adhered to (to distinguish windows from other facade-parts).
  As an example, a further decomposition of a window in transparant and non-transparant parts is disregarded.
  Also, non-decomposed facades are regarded to be non-transparent.
- As can be inspected in fig. 1, SIBE1 defines buildings in a way which is not supported in our data model. Deducting a SIBE1 representation would be very difficult and is not attempted, because the option to specify all buildings as a collection of (non-related) planes offers a way to work around it.
  This has however one very serious drawback: buildings as seperate entity are no longer recognizable in the SIBE1-input, and an affect-status of buildings (an important design decision support aspect) is thus not produced!
- SIBE requires perimeters to be given in a particular application-flavored sequence. It has proved to be an extra burden in the interface development.
- As yet no checks are performed on the positioning of the plane of interest.
- SIBE can handle many planes of interest in one go. This is not supported in the interface because of the ambiguity that would result in determining related and non-related faces in the case of multiple planes of interest.
- Certain standard components in BFEP require a pre-defined sequence numbering of their parts. For example, the "box-shaped room" component requires a strict pre-defined sequence of the six envelope faces. The reason behind this is the fact that the room configuration drives the physical processes (convection) inside the room. The lumped equations which BFEP uses to simulate these processes are configuration-dependent. This problem has been "solved" by adhering to strict conventions during the definition of shape representation objects in ProMod, thus putting no demands on the interface program for finding the right sequence of faces.
  Future extensions will have to allow for more flexibility by performing elaborate operations on the shape-objects to "reconstruct" a BFEP required configuration. This will be most important in the case of decomposed spaces, which is not supported at this moment.
- The previous point is also connected to the unique node-labelling that BFEP uses for component identification and component-links. Benefitting from the proposed decomposition levels, which are identical to the BFEP component hierarchy, the mapping to BFEP-components and exchange-components is handled easily by the interface.
  ALso, no unsurmountable problems are expected in supporting space decomposition. Room-seperating constructions need to be detectable as a topological relation between two adjacent room in that case.
  Another interesting problem is encountered when other relations, e.g. between non-adjacent rooms exist. These type of relations are common in HVAC modelling. This problem can not be solved by the present approach. We would clearly need to be able to define relations which go beyond topological ones.

5. Conclusions and future developments

By way of a simple exercise with two BPE tools, necessary steps to arrive at data integration have been demonstrated.
Based on a sketch of the ideal requirements it was argued that much work in the area of building product definition has still to be done.
It must be recognized also that data integration in itself is not the ultimate goal, but only a first step towards future integrated design systems.

Such design systems could provide some level of intelligent assistance to a designer, if equipped with knowledge of the design process (hence the term process integration by which we denote this additional integration level).
Some of the concepts are presently being explored (be it on a small scale) by adding some design context to the integrated use of BFEP and SIBE.
Fig. 6 presents a snap shot of the user interface of this small aspect-oriented design-tool prototype, which incidentally is not based on ProMod, but has similar capabilities.

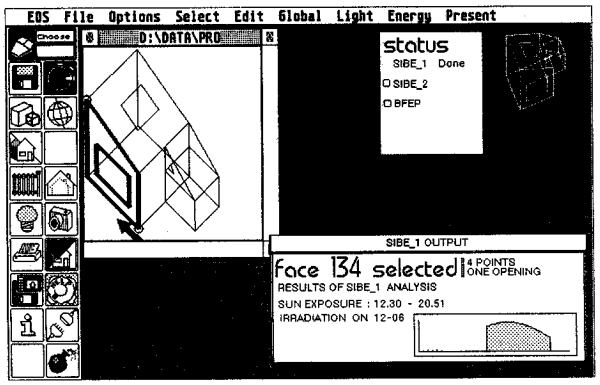

Fig. 6. Snap shot of user interface for a small design tool prototype.

The figure shows the window-oriented design tool interface. The data window shows a model of the design object, serving mainly for viewing and selection (the highlighted wall) purposes.
An output window lets the user view calculated BPE-results, in this case the direct irradiation data for the selected wall.
The status window is for a selection of other BPE tools. By adding more "design intelligence" to the design tool it will be able to help the designer in choosing options, make judgements etc. with respect to solar and thermal evaluations.
The European joint project COMBINE, started in 1990 will combine 7 design tool prototypes (DTP) around one common integrated building data model. The selected DTP's deal with different aspects (e.g. HVAC, energy, design of external building elements, inner space design) in the design stage. The COMBINE project should be able to provide valuable input to future full-blown design system developments. For future details in COMBINE, we refer to (Augenbroe and Winkelmann 1991).

References

Augenbroe, G.L.M., 1986, "Research-oriented Tools for Temperature Calculation in Buildings", Proceedings, Second Int. Conf. on System Simulation in Buildings, Liège.

Augenbroe, G.L.M. and Winkelmann, F.C., 1991, "Integration of Simulation into the Building Design Process", Proceedings IBPSA Conf. on Building Simulation 1991.

Clarke, J.A., 1988, "The Energy Kernel System", Energy and Buildings, Vol. 10, pp. 259-266.

Danner, W.F., 1990, "A proposed Integration Framework for STEP", NIST US Dept. of Commerce, NISTIR 90-4295.

Danner, W.F. and Young, Y., 1991, "Generic Product Data Resources for STEP", NIST, US Dept. of Commerce, NISTIR (draft VS 0.9a).

Gale, R., 1986, "Information Modelling, IDEF-1 and IDEF-1X", Appendix D1 to PDES Initiation Effort Report.

Gieling, W.F., 1989, "General AEC Reference Model (GARM)", IBBC-TNO Delft, Report BI-88-150.

ISO, 1989, External Representation of Product Definition Data (STEP), ISO-DP 102030.

Krom, R.P., 1989, "XPCL, Ontwerp en Implementatie van een Programmeeromgeving voor het modelleren van Produkten", IBBC-TNO Delft, Report B-89-048.

Kuiper, P. et al, 1987, "Beproevingssysteem Produktmodellering Deel 1", CADLAB IBBC-TNO Delft, Report B-87-600.

Tolman, F.B. et al, 1989, "Four years of Product Modelling, collected papers", IBBC-TNO Delft, Report BI-89-140.

Voorden, M. van der, 1985, "Determining the Rate of Solar Irradiation in the Built Environment", Proceedings, Ninth Biennial Congress of ISES, vol. 1, pp 425-430.

Voorden, M. van der, 1988, "Advantages of Non-ray tracing Methodology for Direct Solar irradiation calculation", Proceeding of Int. Conf. North Sun '88, Borlänge, vol. 1, pp 611-615.

Wilschut, P.E., 1990, "Koppeling van twee Bouwfysische Applikaties aan ProMod", TU Delft, Struc. and Building Engineering Group, Master's Thesis Report C2-88-06A.